clear;

clc;

I=imread('pout.tif');

[n,m]=size(I);

I = uint8(I);

Threshold = mean2(I);% mean of matrix elements

done = false;

while ~done

G1 = I > Threshold;

G2 = ~G1;

m1=mean(I(G1));m2=mean(I(G2));

newThreshold = 0.5 \* (m1 + m2);

done = abs(Threshold - newThreshold) >= 1;

Threshold = newThreshold;

end

% get binary image

B= I>Threshold;

B=uint8(B\*255);

%imshow(B);

% Identify which pixels are connected to each other.

% Each group of connected pixels will be given a label, a number, to identify it

[labeledImage , NumOfObjects] = bwlabel(B, 8);

fprintf('number of objects = %d ',NumOfObjects);

Measurements = regionprops(labeledImage, 'all');

%cell store any type similar to type Object (store diff type in one matrix)

objPixels=cell(n\*m,NumOfObjects); %maxmium no of pixels when there is one object fill the whole image

objArea=zeros(1,NumOfObjects);

objPerimeter=zeros(1,NumOfObjects);

for k = 1 : NumOfObjects %for each object

e=Measurements(k).PixelIdxList; %return location pixels of object in memory

[x,y]=ind2sub(size(B), e); %convert to location in matrix B

%store each x[i],y[i] in a matrix in new matrix in the form (x[i],y[i]) for each pixel i in an

%object k

for i= 1: size(x)

S=sprintf("(%d,%d)",x(i),y(i)); %convert to string in the form (x(i),y(j))

objPixels(i,k)={S}; %convert string to cell and add to objpixels

end

objArea(k) = Measurements(k).Area; % Get area.

objPerimeter(k) = Measurements(k).Perimeter; % Get perimeter.

end

%edge detection

J = edge(I, 'prewitt');

%J = edge(I, 'sobel');

subplot 121; imshow(I); title('Original image');

subplot 122; imshow(J); title('edge detection');